Paper

Optimization approach with ρ-proximal convexification for Internet traffic control

Adam Kozakiewicz

Abstract—The optimization flow control algorithm for traffic control in computer networks, introduced by Steven H. Low, works only for concave utility functions. This assumption is rather optimistic and leads to several problems, especially with streaming applications. In an earlier paper we introduced a modification of the algorithm based on the idea of proximal convexification. In this paper we extend this approach, replacing the proximal method with the ρ-proximal method. The new method mixes the quadratic proximal term with higher-order terms, achieving better results. The algorithms are compared in a simple numerical experiment.

Keywords— nonlinear programming, price method, convexification, network control.

1. Introduction

It is possible to formulate the problem of traffic control in the Internet as an optimization problem. Many proposed control methods are based on the idea of solving such a problem using a distributed algorithm based on the dual approach. Recently this approach was even used to model TCP Vegas [4]. However, most of such proposals assume convexity of the problem, eliminating the possibility of a duality gap. The model is then simplified, but not necessarily realistic. Nonconvexity, while definitely not desireable, may be important. There are a lot of methods for convexification of a nonconvex optimization problem, but most of them do not correspond to the structure of the network. In [2] we proposed a way of introducing convexification in the algorithm proposed by Steven H. Low ([1] with a correction in [5]). The method we proposed is not the only one possible and in this paper some modifications are suggested.

In Section 2 we describe the problem, introduce the concept of proximal convexification and the partial convexification algorithm for traffic control from [2]. Section 3 presents the ρ-proximal convexification method, and Section 4 explains how it can be adapted to our problem. Finally, in Section 5 we present some simulation results and conclude in Section 6 with a summary.

Note: We use the term *convex problems* for a class of problems including both minimization of convex functions and maximization of concave functions, both over convex sets. Converting a problem to fit this class is therefore called *convexification*. In the case of a maximization problem, convexification means making the goal function concave (hence the names *convexification parameter* and *center of convexification*).

2. Optimization in traffic control

An often mentioned (e.g., [6–10]) way of dealing with the problem of sharing network resources in the best possible way is maximizing average (or, equivalently, total) utility, as defined by the sources' individual utility functions. The optimal solution of such a problem is *proportionally fair*. This approach is the essence of Steven H. Low's optimization flow control algorithm, presented in [1] (with a correction to the proof in [5]).

The network is modelled as a set of unidirectional links, L with capacities $c_l, l \in L$, used by a set of sources, S. The links may also be bidirectional with the capacity limit on the sum of traffic in both directions. Each source uses a predefined set of links L_s (static routing), although the model can be modified to choose from multiple paths. The relation between sources and links defined by sets L_s can be represented as the binary routing matrix $\mathbf{A}$. Each source's rate x_s is bounded by limits m_s and M_s and derived from the state of the network and its own utility, defined as a function $U_s(x_s)$, preferably zero for $x_s = m_s$ and increasing. Utility may be normalized (to be 1 at $x_s = M_s$) or not. Capacities and rates can be expressed as vectors $\mathbf{c} = [c_l]_{l\in L}$, $\mathbf{x} = [x_s]_{s\in S}$. For feasibility it is necessary that for each link the sum of lower limits of sources using this link be lower than its capacity. To eliminate the possibility of a duality gap, strong concavity of the utility function for each source is also necessary. Combining both assumptions leads to certain difficulties, explained later.

The problem in this case can be formulated as follows:

$$\begin{aligned} &\max_{\mathbf{x}\in I} \sum_{s\in S} U_s(x_s) \\ &\text{subject to } \mathbf{Ax} \le \mathbf{c} \\ &I = \bigtimes_{s\in S} I_s \\ &I_s = [m_s, M_s] \end{aligned} \tag{1}$$

and it can be solved by application of the dual approach.

Using prices (Lagrange multipliers) to communicate the state of the network to sources it is possible to distribute effectively the computations. A simple, synchronous algorithm follows:

Algorithm 1 (synchrononous gradient projection)
At times $t = 1, 2, \ldots$:

- Link (router) l:
  1. Collects the rates x_s of all sources using the link ($s \in S_l$) and computes $x^l = \sum_{s \in S_l} x_s$.
  2. Computes a new price $\xi_l(t+1) = [\xi_l(t) + \gamma(x^l(t) - c_l)]^+$, where $\gamma > 0$ is a stepsize, and $[z]^+ = \max\{z, 0\}$.
  3. Communicates new price $\xi_l(t+1)$ to all sources that use link l.
- Source s:
  1. Receives from the network the total price along its path $\xi^s(t) = \sum_{l \in L_s} \xi_l(t)$.
  2. Computes its new rate $x_s(t+1) = \arg\max_x [U_s(x) - \xi^s(t)x]$.
  3. Communicates its new rate to all links on its path.

If γ is small enough and the problem is convex, this algorithm is convergent, even with modifications, ranging from asynchronous computation to multiple paths and limited communication ([1] and later papers). In reality, however, for many applications the utility function is unlikely to be concave (a good explanation can be found in [7]). In most cases this means, that utility grows rapidly around some preferred transmission rate, much higher than the minimum, and only slowly grows for higher rates. Eliminating such nonconcavities by raising lower limits to the "preferred" rates and using convex approximations, as suggested in [1], may lead to exceeding the limit defined in the feasibility assumption and other problems, explained in [2].

2.1. Proximal convexification

There are many available methods of convexification (a survey can be found in [3]). The augmented Lagrangian method [11, 12], although simple and effective, would destroy separability and the link prices would not be sufficient information for the sources. The proximal method (see [13]) deals with nonseparability at the cost of an additional level of iteration.
A nonconvex problem:

$$\begin{aligned} &\min_x \ f(x) \\ &\text{subject to } h(x) = 0, \quad g(x) \le 0, \end{aligned} \tag{2}$$

where all functions are twice continously differentiable, can be solved by iterative application of the multipliers method to a modified problem:

$$\begin{aligned} &\min_x \ f(x) + (\theta/2)\|y - x\|^2 \\ &\text{subject to } h(x) = 0, \quad g(x) \le 0. \end{aligned} \tag{3}$$

where y is a parameter approximating the optimal point and $\theta > 0$ is the convexification parameter. Large values of θ give a larger area of convergence, but its rate becomes slower. At a higher, additional level of iteration we treat the value of the optimised function in the above problem depending on the parameter y, $\phi(y)$ as the goal function and solve $\min_y \phi(y)$ using some simple iterative algorithm, usually a Jacobi type iteration $y_{k+1} = \hat{x}_k$, where $\hat{x}_k$ is the solution of (3) for $y = y_k$. The method can easily be adapted for problems with inequality constraints.

2.2. Partial convexification

The additional level of iteration cannot be implemented in a changing distributed environment. Finding the right value for the convexification parameter – not too large, not too small – is difficult too. Luckily, the strict convergence guarantees of the original method are not necessary for traffic control, neither is strict optimality, as long as "good" solutions are found. In a real network the algorithm may never have enough time to converge to a fixed solution – the state of the network changes all the time, even during optimization. Old connections terminate, new ones are created and even an existing connection may have changing requirements – the problem is not a stationary one. More important than optimality are nonoscillatory behaviour, ability to smoothen perturbations and a possibility of simple, efficient implementation. Therefore in [2] we proposed the following algorithm, merging the higher iteration levels and dropping the requirement for the convexification strength, represented by a parameter θ, to be sufficiently large to make the problem convex:

Algorithm 2 (convexified projection algorithm)
At times $t = 1, 2, \ldots$:

- Link (router) l (as in Algorithm 1):
  1. Collects the rates x_s of all sources using this link ($s \in S_l$) and computes $x^l = \sum_{s \in S_l} x_s$.
  2. Computes a new price $\xi_l(t+1) = [\xi_l(t) + \gamma(x^l(t) - c_l)]^+$, where $\gamma > 0$ is a stepsize, and $[z]^+ = \max\{z, 0\}$.
  3. Communicates new price $\xi_l(t+1)$ to all sources that use link l.
- Source s:
  1. Receives from the network the total price on its path $\xi^s(t) = \sum_{l \in L_s} \xi_l(t)$.
  2. Computes its new rate $x_s(t+1) = \arg\max_x \left[U_s(x) - \xi^s(t)x - (\theta/\rho)\,\|x_s(t) - x\|_\rho^\rho\right]$, where $\theta > 0$ and $\rho \ge 1$ are parameters.
  3. Communicates new rate to all links on its path.

Parameter ρ in this algorithm is equal to 2 by default, other values, although possible, are not recommended, as they additionally weaken the local convexification effect. Parameter θ should actually be a source-dependent value θ_s, so we use the parameter $\eta = \theta_s |M_s - m_s|^\rho$ as a global specification of convexification strength.

3. The ρ-proximal convexification

The ρ-proximal convexification method is a version of the proximal algorithm, developed in [3, 14]. It modifies the convexifying terms added to the goal function of the modified problem (3) in an attempt to attain faster convergence. The modified problem in the new version is as follows:

$$\min_x \; f(x) + (\theta/2)\|y-x\|^2 + (\theta/\rho)\|y-x\|_\rho^\rho \quad (4)$$
$$\text{subject to } h(x) = 0, \quad g(x) \le 0.$$

It is possible to weigh the two convexifying terms differently, but the general idea remains the same. It is important that the quadratic term is not completely removed, as it is necessary for local convexity near y, not provided by higher order terms. Value ρ should be greater than 2. For practical reasons integer values are preferred. The most probable choice is 4, as it is easy to compute from the quadratic term, 6 or 8 might also be used for more difficult problems. For further analysis see [3, 14].

4. Modified partial convexification

The modification proposed in Section 3 can also be applied to Algorithm 2, giving the following algorithm:

Algorithm 3 (modified convexified projection algorithm)
At times $t = 1, 2, \ldots$:

- Link (router) l (as in Algorithm 1):
  1. Collects the rates x_s of all sources using this link ($s \in S_l$) and computes $x^l = \sum_{s \in S_l} x_s$.
  2. Computes a new price $\xi_l(t+1) = [\xi_l(t) + \gamma(x^l(t) - c_l)]^+$, where $\gamma > 0$ is a stepsize, and $[z]^+ = \max\{z, 0\}$.
  3. Communicates new price $\xi_l(t+1)$ to all sources that use link l.
- Source s:
  1. Receives from the network the total price on its path $\xi^s(t) = \sum_{l \in L_s} \xi_l(t)$.
  2. Computes its new rate

$$x_s(t+1) = \arg\max_x \Big[U_s(x) - \xi^s(t)x - \theta\Big((\alpha/2)\,\|x_s(t) - x\|_2^2 + ((1-\alpha)/\rho)\,\|x_s(t) - x\|_\rho^\rho\Big)\Big],$$

  where $\theta > 0$, $\alpha \in (0,1)$ (preferably not too small) and $\rho \ge 2$ are parameters.
  3. Communicates new rate to all links on its path.

The newly introduced parameter α can be chosen arbitrarily. For $\alpha = 0$ or $\alpha = 1$ the algorithm is identical to Algorithm 2. In the simulations we decided to make α a per-source parameter α_s, derived from the assumption that, when the argument is $M_s - m_s$, the quadratic term is four times smaller than the other one. This way near the center of convexification ($x(t)$ in this algorithm) the quadratic term still dominates, while at longer range the stronger convexifier is more important. Parameter θ is chosen the same way as before, ρ defaults to 4. The generally better convexification of the ρ-proximal method suggests, that the new algorithm may be more effective than the original.

The mixed quadratic – higher order concave term has the advantages of both the original and convexified algorithm. As in the Algorithm 2, there is a mechanism to prevent the sources from oscillatory behaviour – the higher order term adds a penalty for such big changes. At the same time, the quadratic term is smaller and doesn't affect the utility function more than necessary. This is similar to the simple convexified algorithm with ρ set to any value greater than 2. This method however does not completely eliminate the quadratic term, which results in better local concavity near $x_s(k)$. Through analogy to the family of proximal convexification methods for general optimization there are reasons to believe, that this may give better results for small steps. The simple convexified algorithm with $\rho \ge 3$ tended to cause oscillations for some types of utility functions and, although better than the original one, often gave worse results than with $\rho = 2$. The additional quadratic term is added to eliminate that effect.

The new algorithm is more complicated, but not significantly so. Adding one more multiplication (two, including a constant) and one addition doesn't have much effect on the calculation time for the utility function. The only real problem is the maximization of utility. In the original algorithm it was done by reversing the function. In the convexified algorithms, including the simple one, such a solution, while possible and suggested, is difficult to find. The convexification makes utility a function of two variables. Luckily only one of them is maximized, the other one is known in each iteration. For simplicity an optimization procedure was used in the experiment instead of an inverse function.

5. Computational results

The algorithms were tested on a single link with four sources, described by four different, S-shaped utility functions. For all functions $U(m_s) = 0$, $m_s = 1$ (not 0, to simulate the trickle of packets necessary to keep the connection to the router and identify the current price), $U(M_s) = 1$ and $M_s = 100$, except $M_4 = 120$. Another parameter is the rate at which utility is equal to 0.5, for these four sources those rates are 40, 60, 50 and 100. The last source also has a steeper gradient at this point, simulating a large, nonelastic connection.

We also ran tests with concave approximations of these functions, modifying the lower bound on sources' rates accordingly (if our algorithms failed in the simple convex case, their effectiveness against nonconvexity wouldn't matter). Tests were repeated with different capacities of the link and different parameters of the algorithms (Fig. 1). We set γ to 10^{-5}. Note, that for convex approximations the minimal bandwidth required for feasibility is about 190 units, so a problem with bandwidth of 150 units, more than enough to support any of the sources at full utility, does not have a feasible solution after the approximation. We will look at some of the results for a bandwidth of 150, 250 and 400 units. In the first case there is no feasible solution for the concave approximations, and the nonconvexity is significant. In the second case the approximation will work, but the original problem's nonconvexity can't be ignored. In the third case there is almost enough bandwidth and the nonconvexity should not have any effect on the calculation.

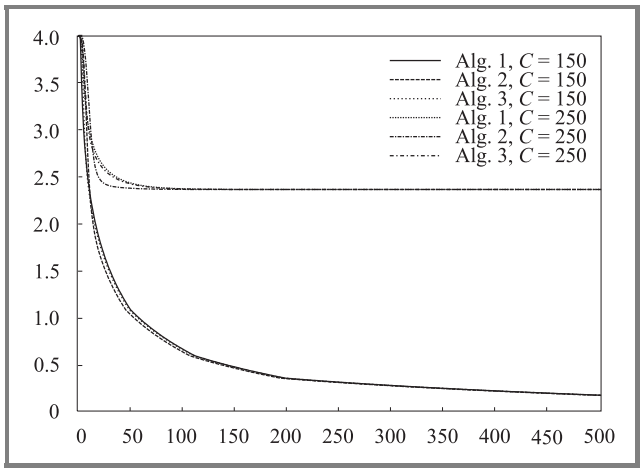


Fig. 1. The total utility functions for the concave approximation with link capacity of 150 and 250 units, different algorithms.

We analyzed the following results: the rates of all sources, the total utility of the sources, the price of the link and the difference between the total traffic and available capacity. Only some of those are shown and we concentrate on the total utility. The convexification parameter η is set to 2. Our tests have shown, that a value of 0.5 is enough for the convexification to be useful and reduce oscillations of the original algorithm, but 2 is better, as it is enough for the algorithm to converge to a stable solution. Another observation is that with little congestion ($C = 400$) the nonconvexity is not important, unless there is a huge number of small connections. The graphs in this case were almost identical, whether the original utility functions, or their concave approximations were used. Because of that we only present the graph for the nonconvex case (Fig. 2) – in the convex case all processes were about 20% slower, but very similar in shape, and total utility converged to 3.8891 instead of 3.9189, minor differences due to approximation error.

5.1. Concave approximation

As we can see in Figs. 1 and 2, all algorithms work well for the concave approximation, if a feasible solution exists. The convexified algorithms tend to react a little slower and overreact, causing a short oscillation – this is a predictable effect of the introduced inertia. The graph for the largest capacity shows the first advantage of the ρ-proximal algorithm over the quadratic one. Its reduced local effects reduce the oscillation and its results are close to the original algorithm's. The quadratic convexification doubles the time required to reach stable state. For $C = 250$ however, the proximal algorithm converged faster, although it reacted last to the changing price.

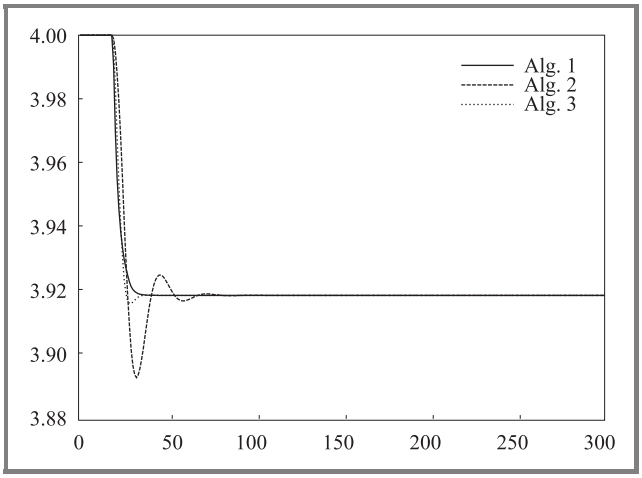


Fig. 2. The total utility functions for the nonconcave case with link capacity of 400 units, different algorithms. The graph for the concave approximation is very similar.

When no feasible solution exists, the total utility decreases exponentially, almost identically for all algorithms (Fig. 1, $C = 150$). This reduction corresponds to the logarithmic shape of the approximation and a constant, linear growth of the price, caused by an almost constant gap between the capacity (150) and the traffic, nearing the sum of lower bounds (about 191). When the price reaches the gradient of the total utility at the lower bound, the utility will achieve 0 and stay there, with the price growing idefinitely.

5.2. Original problem

With congested links and nonconcave utility functions the problem is more difficult. For both $C = 150$ and $C = 250$ there are feasible solutions if convex approximation is not used, but the original algorithm, as proposed by Steven H. Low does not reach one – instead, it oscillates indefinitely (Fig. 3). This confirms the expected behaviour for nonconvex systems. A smaller value of γ will not solve this problem, the algorithm will not converge. While the average utility will be near optimum, the oscillations may not be acceptable.

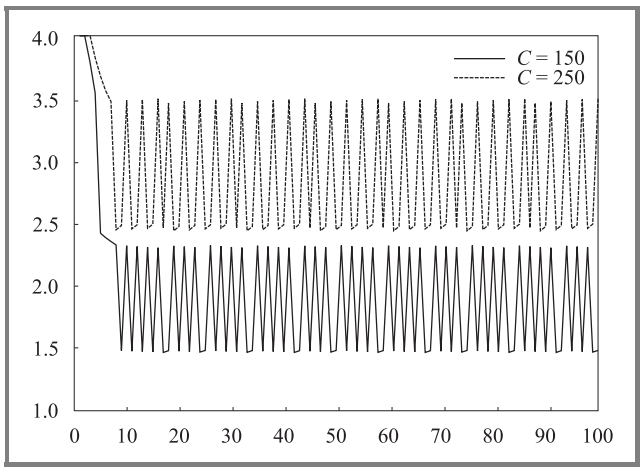


Fig. 3. The total utility functions for the concave approximation with link capacity of 150 and 250 units, Algorithm 1.

The convexified algorithms all converge to a solution (Fig. 4). The initial jumps are the effect of changing the convexification center to near 0 – the convexified function rapidly changes shape and the price has to be corrected. After one or two such rapid changes the algorithms converge. The discontinuities are less prominent with more available bandwidth, as the prices change less rapidly in this case.

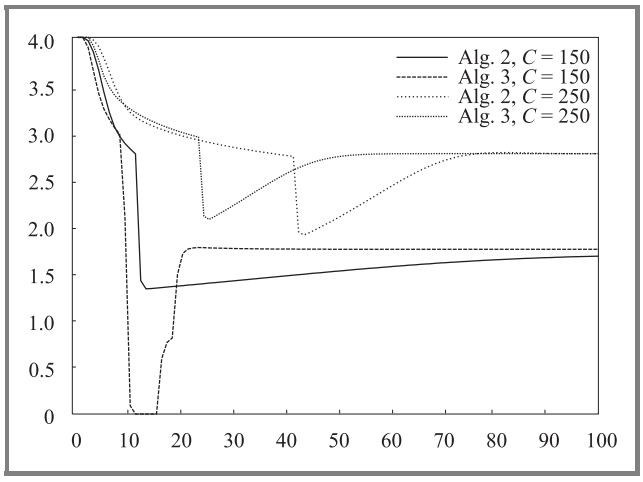


Fig. 4. The total utility functions for the concave approximation with link capacity of 150 and 250 units, convexified algorithms.

These graphs show the advantages of the ρ-proximal mode: the fourth-power term only works further from the convexification center (compared to the quadratic one), so as it moves to lower values, the convexification weakens and a jump occurs sooner. The local convexification with the quadratic term is weaker than in the quadratic-only version, so the algorithm converges faster. It is a general rule for convexification methods – the weaker the convexification, the faster the convergence, unless it is too weak, in which case the algorithm does not converge at all. The ρ-proximal algorithm is therefore the fastest one in this test. The quadratic term is unnecessarily slowed down, but also converges well.

5.3. Other results

The utility is not the only result we collected. In this section we will show, how it corresponds to more physical values. We will now focus on a system with 250 units of capacity and nonconcave utility functions, controlled by Algorithms 1 and 3. Its total utility can be found on Figs. 3 and 4. The sources' rates, price on the link, and the link's overuse (the difference between offered load and capacity) are depicted in Figs. 5, 6 and 7.

The Low's algorithm fails as soon as the price crosses a critical value for source number 4, which immediately jumps to its lower bound (an effect of nonconvexity). It then switches back and forth between the two ranges of rates, with price changing accordingly, as the total offered load is either significantly below or over the capacity.

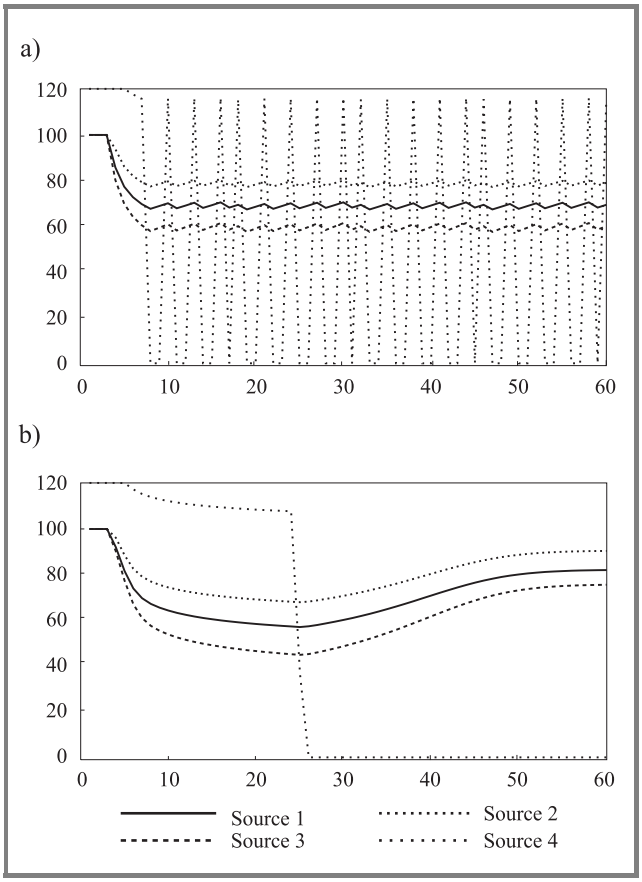


Fig. 5. Traffic rates for $C = 250$ and nonconvex utility functions: (a) Low's algorithm (Alg. 1); (b) ρ-proximal algorithm (Alg. 3).

Our ρ-proximal method also reaches the same point, a bit later, as the convexification slows it down. When the fourth source reduces its rate to minimum, however, it keeps that value, while the others increase their offered traffic to use up the free capacity, just as it should be. The price (Fig. 7) changes accordingly: first it grows to 0.0166 to defeat con-

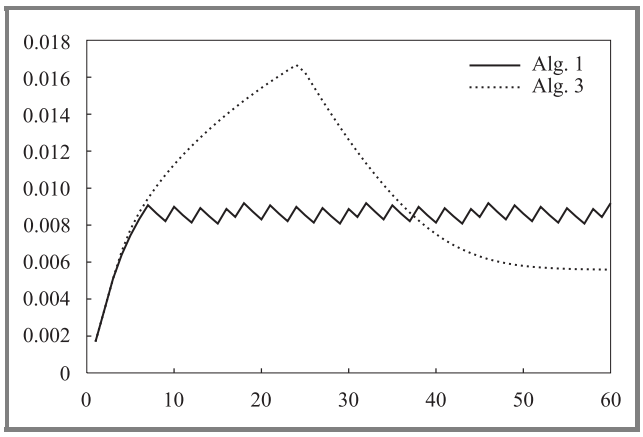


Fig. 6. Link price for $C = 250$, nonconvex utility functions, algorithms: Low's and ρ-proximal.

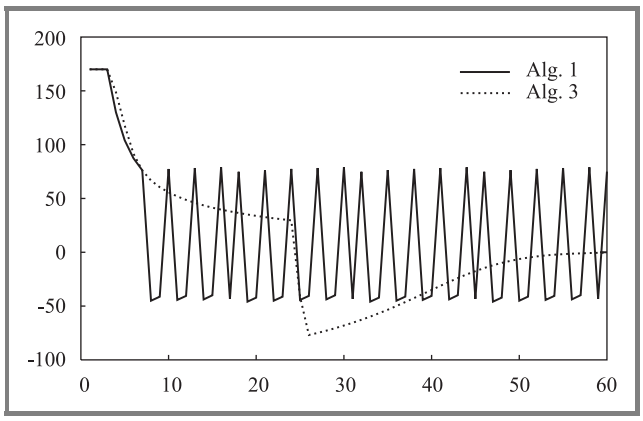


Fig. 7. Excess traffic/slack for $C = 250$, nonconvex utility functions, algorithms: Low's and ρ-proximal.

vexification, then, when the traffic suddenly drops and convexification keeps source 4 at low traffic rate, it returns to the proper level of 0.0056. Figure 7 shows how the excess traffic drops to the point where one of the connections has to switch to minimum traffic. Afterwards there is quite a lot of slack, and traffic grows to fill it.

6. Summary

In this paper we present the results of a new experiment with the convexified gradient projection algoritm and introduce a modification of this technique using ρ-proximal convexifying term.

The experiment for the older, quadratic variant confirmed the results of [2], using a very different approach – a single link with few, easily observable connections, instead of a network of 11 nodes with hundreds of connections and analysis limited to statistics. It is at the same time a complete reimplementation.

The new ρ-proximal variant proved to be as effective in stopping oscillations as the quadratic one and accellerates the convergence. Both work well for nonconcave utility functions, where the original Algorithm 1 fails.

Acknowledgment

This work was partially supported by a stipend from the Foundation for Polish Science.

References

[1] S. H. Low and D. E. Lapsley, "Optimization flow control. I: Basic algorithm and convergence", *IEEE/ACM Trans. Netw.*, vol. 7, no. 6, pp. 861–874, 1999.

[2] A. Kozakiewicz, A. Karbowski, and K. Malinowski, "Partial convexification and its application to Internet traffic control", in *Proc. 8th IEEE Int. Conf. MMAR'2002*, Szczecin, Poland, vol. 1, pp. 275–280, 2002.

[3] A. Kozakiewicz, "Zastosowanie uwypuklania w optymalizacji wielokryterialnej, hierarchicznej i globalnej". Warszawa: Politechnika Warszawska, 2001 (M.Sc. thesis in Polish).

[4] S. H. Low, L. Petersen, and L. Wang, "Understanding Vegas: a duality model", *J. ACM*, vol. 49, no. 2, pp. 207–235, 2002.

[5] A. Karbowski, "Comments on optimization flow control. I: Basic algorithm and convergence", *IEEE/ACM Trans. Netw.*, vol. 11, no. 2, pp. 338–339, 2003.

[6] F. P. Kelly, A. K. Maulloo, and D. K. H. Tan, "Rate control for communication networks: shadow prices, proportional fairness and stability", *J. Oper. Res. Soc.*, vol. 49, pp. 237–252, 1998.

[7] P. Key and L. Massoulié, "User policies in a network implementing congestion pricing", in *Worksh. Internet Serv. Qual. Econom.*, Cambridge, USA, 1999.

[8] P. Key, D. McAuley, P. Barham, and K. Laevens, "Congestion pricing for congestion avoidance", Microsoft Res. Tech. Rep. MSR-TR-99-15, 1999.

[9] R. J. La and V. Anantharam, "Charge-sensitive TCP and rate control in the Internet", in *Proc. Infocom'2000*, Tel Aviv, Israel, 2000, pp. 1166–1175.

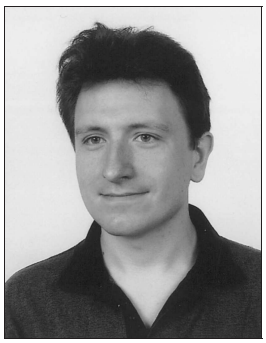
[10] D. Wischik, "How to mark fairly", in *Worksh. Internet Serv. Qual. Econom.*, Cambridge, USA, 1999.

[11] D. P. Bertsekas, *Constrained Optimization and Lagrange Multipliers Methods*. London: Academic Press, 1982.

[12] D. P. Bertsekas, *Nonlinear Programming*. 2nd ed. Belmont: Athena Scientific, 1999.

[13] D. P. Bertsekas, "Convexification procedures and decomposition methods for nonconvex optimization problems", *J. Opt. Theory Appl.*, vol. 29, no. 2, pp. 169–197, 1979.

[14] A. Kozakiewicz and A. Karbowski, "Metoda ρ-proksymalna uwypuklania zadań optymalizacji", in *Proc. 4th Nat. Conf. MSK'2003*, Kraków, Poland, 2003, pp. 343–346 (in Polish).



Adam Kozakiewicz obtained his M.Sc. degree at the Warsaw University of Technology in 2001 and is currently a Ph.D. student there. Student member of IEEE. Interests include nonlinear optimization, distributed computation, grid technology, modelling of computer networks, pricing and traffic control mechanisms.

e-mail: akozakie@ia.pw.edu.pl
Institute of Control and Computation Engineering
Warsaw University of Technology
Nowowiejska st 15/19
00-665 Warsaw, Poland